Freeform Search

Database: US Pre-Grant Publication Full-Text Database
US Patents Full-Text Database
US OCR Full-Text Database
EPO Abstracts Database
JPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Term: L7 and 16

Display: 10 **Documents in Display Format:** - **Starting with Number** 1

Generate: ○ **Hit List** ◉ **Hit Count** ○ **Side by Side** ○ **Image**

Search Clear Interrupt

Search History

DATE: Thursday, September 16, 2004 Printable Copy Create Case

Set Name side by side	Query	Hit Count	Set Name result set
DB=PGPB,USPT,USOC,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR			
L8	L7 and l6	1	L8
L7	(compress$ same (record$ near5 level$ or file near5 level$) same (field$ or table$)).ab.	8	L7
L6	(compress$ same (record$ near5 level$ or file near5 level$) same (field$ or table$)).clm.	5	L6
L5	L4 and l3	9	L5
L4	(compress$ same (record$ near5 level$ or file near5 level$)).clm.	63	L4
L3	(compress$ same (record$ near5 level$ or file near5 level$)).ab.	231	L3
L2	(compress$ same (record$ near5 level$ or file near5 level$)).ti.	30	L2
L1	compress$ same (record$ near5 level$ or file near5 level$)	1414	L1

END OF SEARCH HISTORY

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#
End of Result Set

Generate Collection Print

L8: Entry 1 of 1 File: USPT Mar 2, 1999

DOCUMENT-IDENTIFIER: US 5878125 A
TITLE: Method for storing analysis data in a telephone exchange

Abstract Text (1):
The invention relates to a method for storing analysis data, especially digit analysis data in a telephone exchange (21). The data is stored as a tree-like hierarchical data structure that returns the desired result on the basis of given initial data, said data structure comprising records (11; 31; A . . . F) located in several different levels, each record comprising a predetermined number of fields (0 . . . 9, a . . . f), the uppermost level composing the root level of the structure and a single field of a record either being empty or comprising a pointer that points to the destination that may be either a record or a result in a lower level. For enabling new services of a telephone network, such as free numbering, without a continuously increasing need for memory capacity or purchases of devices, stored data is compressed by searching in a single level of the structure all the records the corresponding fields of which are pointing to the same destination, by deleting all except one of the found records and by changing the pointers of upper levels that have pointed to the deleted records so that they point to said one record, whereby said measures are started from the lowest level of the structure, after which they are repeated in the following level if necessary, until the root level is reached.

CLAIMS:

1. A method for storing analysis data, especially digit analysis data in a telephone-exchange (21), said data being stored as a tree-like hierarchical data structure that returns the desired result on the basis of given initial data, the data structure comprising records (11; 31; A . . . F) in several different levels, the records each comprising a predetermined number of fields (0 . . . 9, a . . . f), the uppermost layer composing a root level of the structure and a single field of a record either being empty or comprising a pointer pointing to a destination that may be either a record located in a lower level or the result, characterized by compressing the stored data by searching in a single level of the structure all the records the fields of which corresponding to each other point to the same destination, by deleting all except one of the found records and by changing the pointers of upper levels that have pointed to the deleted records so that they point to said one record, whereby said procedures are started from the lowest level of the structure, after which they are repeated if necessary in the following level until the root level is reached.

Previous Doc Next Doc Go to Doc#

Previous Doc Next Doc Go to Doc#
First Hit

Generate Collection

L2: Entry 3 of 30 File: TDBD Feb 1, 1994

TDB-ACC-NO: NA940287

DISCLOSURE TITLE: Utilizing Drive Level Compression through Cross-File Compressing

PUBLICATION-DATA:
IBM Technical Disclosure Bulletin, February 1994, US

VOLUME NUMBER: 37
ISSUE NUMBER: 2A
PAGE NUMBER: 87 - 88

PUBLICATION-DATE: February 1, 1994 (19940201)

CROSS REFERENCE: 0018-8689-37-2A-87

DISCLOSURE TEXT:

To increase the capacity of optical drives, compression ability can be added. The natural approach to utilizing this would be for the file system to compress the data of an individual file and then store it as it would before. This disclosure covers alterations to the file system structure that allows data of multiple files to be compressed together and still maintain the capability of individual delete of the files. This method enables multiple small files to be written together with a single disk operation and still be compressed. Efficiency in compressing small files is gained while maintaining the existing file system semantics. - The unit of compression by the drive is termed a "transfer unit". A transfer unit is written to the drive. The drive then compresses that buffer and writes it to the media beginning at the designated address and returns the resulting size of the compressed write to the issuing system. Without modification of the current file system, a single files data may consist of multiple transfer units, but a transfer unit would contain the data of only one file. The modification that follows allows a transfer unit to contain the data of multiple files. - Currently the IBM High Performance Optical File System format contains a directory with an entry for each file. This entry uses extent descriptors to point directly to the data of the file. A new structure is inserted "between" these extent descriptors and the data. Instead of pointing directly to physical sectors, the directory entry will point to a "compression map" that is used to store the physical location of transfer units on the media. - The compression occurs during packed writes. In this scenario, multiple files are being written with single writes already, hence the cross file cache buffer contains the data of the transfer unit. When allocating space for a packed write scenario, the code allocates a physical area as before. The buffers are written sequentially to this area using the drive compaction. Unused area saved by the compaction can then be placed back into the free space file. - A compression map consists of an array of "transfer unit descriptors" (TUD) and a bitmap array that will be used to map deleted logical blocks. The transfer unit descriptor consists of three fields and is used to map a range of logical blocks (original user data) to a transfer unit on the disk that is of the same size or smaller: logical size - Size of the original user data. start address - Physical block of start of transfer unit on disk. physical size - Physical size of the transfer unit on the disk. - Since the compression map is an array of the transfer unit

descriptors, it maps a large range of logical blocks (user data) to a series of transfer units on the disk. The bitmap is covered with delete operations below. - In the directory entry, the extent descriptor now used to point directly to the data at a physical location will be replaced by a "Compressed Extent Descriptor" (CED). This structure maps the logical data of the file to some logical block range of a compression map. The system then refers to the compression map to determine which transfer unit(s) contain the files data. The CED contains: compression map identifier A field used to locate a compression map. This could be a physical address containing a map or an index or key into another structure such as a system file containing all the compression maps for a file system. logical block start The logical block "into" the compression map that the extent represented by this CED begins at. logical extent size The size in logical blocks of the extent represented by this CED. - Once the compression map structure is defined, the read and write algorithms are straight forward. Delete operations however require more definition. - Upon delete of a file, the space occupied by the file cannot always be directly released to the system since the file may be contained within a transfer unit with other undeleted files. When deleting a file, the bitmap of the compression map is set to indicate that certain logical blocks of the map are now free. If all the logical blocks of a transfer unit are then free, the TUD is replaced with a skip descriptor to act as a logical block range place holder and the physical space is free back to the system. The skip descriptor is a TUD with a valid logical size field but start and physical size fields set to a special value to indicate that this is only a place holder. - Should the system ever run out of space, a garbage collection operation could be performed to collect space.

Previous Doc Next Doc Go to Doc#